[[1]](#footnote-2)

*Overview*—In lieu of an abstract, use this section to 1) describe the project, 2) explain your methodology, 3) explain the outcomes. Shoot for between 200-300 words.

Assignment 1: Calculator Application

Luke Brandon

The project was to create a calculator app that implements the basic functions of a calculator in Android Studio. This was accomplished using an object oriented design approach and a central

# INTRODUCTION

The goal of this assignment was to create a working calculator application in Android Studio that is capable of handling addition, subtraction, multiplication, division, floating point numbers, negative numbers, a clear operation that would clear what is inputted, and a backspace operation that would backspace the furthest right character on the screen.

# Application Design

The calculator app, given its simplicity, does not implement a common design ideology such as MVC or MVP however the application does use an object oriented design approach rather than just generating a string that is parsed for the operations and does the calculations in that way as well as some features that enhance the user experience and the user’s knowledge of the application’s operation.

## Object Oriented Design

The object oriented design allows the actual parsing of the expressions to be much simpler as Terms (numbers) and Operators (+,-,\*,/) behave differently and have different functions. Each of these objects implement their own functions, operations (such as backspacing, this operation looks different to a Term or Operator), and conversions which allows for there to be less logic in the MainActivity and also allows for better separation of code, meaning that the code that is closely related (ex: code for Terms) to be in the same place. The “Model” in the case of an MVP or MVC design is most closely related to the ArrayList of ExpressionComponents (An abstract class which both Term and Operator inherit from) which stores their entire expression in the order inputted by the user. This design makes distinguishing Operators and Terms very easy making specific cases (for example, if the user tries to put one operator directly after another) as there are many cases that need to be avoided or checked for in a calculator. This design also makes this app much easier to extend in to a much complex calculator as an object based design is almost entirely necessary for an advanced calculator.

B. *Honesty and Clarity Focus*

To me, there is nothing more annoying than not knowing that an application is hiding something from you or blocking you from doing something without you knowing, that is why this calculator application was developed with honesty and clarity in mind. The main expression of this is the Error field that is used in the application. This field tells the user when they are blocked from doing something and what they are doing wrong

Consider the following examples:

* 2+\*

This does not make any sense to plug into a calculator, you cannot have one operate directly after another, so, in this case, the application will tell the user that they “Expressions should not end with an operator”.

* 2+ EQUALS

This also does not make any sense as the trailing operator is not going to be used and nothing will compute, thus the application would prompt the user that this is “Invalid input”.

# Results

This section should describe how your application performs. For class projects, you should describe how it compares to the written test objectives. Show an image or two of your screen. You may consider using Adobe Illustrator/Inkscape/Paint/Windows Photo Editor to mark up the images to point out particular buttons, components, or other things that you may want to call to attention. This should mainly focus on does the application work as intended, and as described in the introduction and the application design sections.

The calculator works as assigned with some extra features and rigidity.

\*PICTURES\*

# Future Work

For class projects, if you run into bugs in your application that you haven’t fixed, include them here with a test case. If they fail one of the hidden tests, but you explain what you think is happening and that you identified the bug, I may take fewer points off. It is good practice to be used to explaining where your app architecture and code falls short so that you can get help in improving it.

# Discussion

Optional section where you may include any relevant details that you didn’t think fit in other sections.

1. [↑](#footnote-ref-2)